(19) Canadian Intellectual Property Office / Office de la Propri,t, Intellectuelle du Canada
An Agency of Industry Canada / Un organisme d'Industrie Canada

(11) CA 2 286 534 (13) A1

(40) 18.04.2001
(43) 18.04.2001

(12)

(21) 2 286 534

(51) Int. Cl.6: H04L 12/22, H04L 9/32

(22) 18.10.1999

(71) AMERICAN GEM CORPORATION,
5000-150 Eglinton E., TORONTO, O1 (CA).

(72) KHANINE, DMITRI (CA).

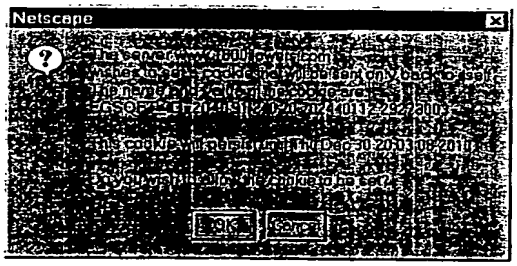
(74) FREEDMAN, GORDON

(54) METHODE DE PROTECTION DE L'ACCES DES UTILISATEURS A DE MULTIPLES FICHIERS PROTEGES ACCESSIBLES PAR RESEAU

(54) METHOD FOR SECURE USER ACCESS TO MULTIPLE NETWORK ACCESSIBLE SECURE FILES

(57)

²²²A method of identifying a session formed between a client system and a network²accessible server system for providing secure access to multiple network ²accessible secure files²is disclosed. According to the method, a request including a session ²identifier an a hashed value²reflective of the session identifier is transmitted to a server. The server ²then transforms the²session identifier using a one-way transformation and the result is compared ²to the received²hashed session identifier. When the comparison result is indicative of a match ²within²predetermined limits, processing the request to provide results and ²transmitting the results to the²client. In an embodiment, the transform is a non-reversible transform provided ²by a neural²network. Optionally, the neural network is modified at intervals in order to ²reduce a risk that²someone will decode the transform technique.²



OPIC
OFFICE DE LA PROPRIÉTÉ INTELLECTUELLE DU CANADA

CIPO
CANADIAN INTELLECTUAL PROPERTY OFFICE

(12)(19)(CA) Demande-Application

(21)(A1) 2,286,534

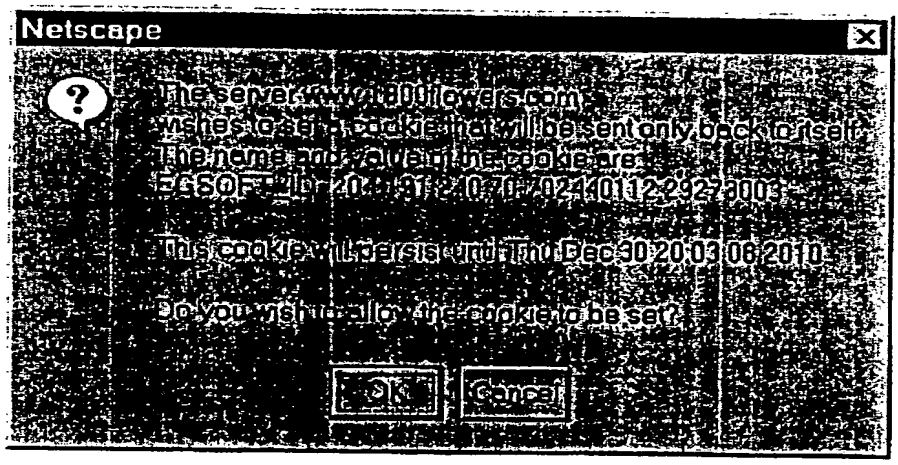
(22) 1999/10/18
(43) 2001/04/18

(72) KHANINE, DMITRI, CA
(71) AMERICAN GEM CORPORATION, CA

(51) Int.Cl.[6] H04L 12/22, H04L 9/32

(54) METHODE DE PROTECTION DE L'ACCES DES UTILISATEURS A DE MULTIPLES FICHIERS PROTEGES ACCESSIBLES PAR RESEAU

(54) METHOD FOR SECURE USER ACCESS TO MULTIPLE NETWORK ACCESSIBLE SECURE FILES



(57) A method of identifying a session formed between a client system and a network accessible server system for providing secure access to multiple network accessible secure files is disclosed. According to the method, a request including a session identifier an a hashed value reflective of the session identifier is transmitted to a server. The server then transforms the session identifier using a one-way transformation and the result is compared to the received hashed session identifier. When the comparison result is indicative of a match within predetermined limits, processing the request to provide results and transmitting the results to the client. In an embodiment, the transform is a non-reversible transform provided by a neural network. Optionally, the neural network is modified at intervals in order to reduce a risk that someone will decode the transform technique

Abstract of the Disclosure

A method of identifying a session formed between a client system and a network accessible server system for providing secure access to multiple network accessible secure files is disclosed. According to the method, a request including a session identifier an a hashed value reflective of the session identifier is transmitted to a server. The server then transforms the session identifier using a one-way transformation and the result is compared to the received hashed session identifier. When the comparison result is indicative of a match within predetermined limits, processing the request to provide results and transmitting the results to the client. In an embodiment, the transform is a non-reversible transform provided by a neural network. Optionally, the neural network is modified at intervals in order to reduce a risk that someone will decode the transform technique.

Method for secure user access to multiple network accessible secure files

Field of the Invention

The present invention relates generally to network communications and more particularly to a method of providing secure access to a plurality of different secure files via an unsecured communication network without requiring further security measures between access to different files.

Background of the invention

The Internet is becoming an extremely prolific form of communication, business, and entertainment. In order to achieve this widespread usage, the Internet has maintained a completely unsecure environment. In such an environment, communications can be intercepted and retransmitted. Further, one user with sufficient know how can masquerade as another user with little difficulty.

In order to provide business services on the internet, it is essential that some form of secure communication and/or secure data access exists. For example, an Internet based catalogue might require an individual to call in their orders in order to ensure that security is maintained. Obviously, this limits the power of the Internet in terms of ease of use and documentation.

Another form of Internet based business relates to accessing information over the network. Internet based stock market trading exists as do other commercial sites selling information or information based services. A problem with these sites is that it is preferable to provide different information on different pages – within different accessible files. That said, it is also preferred that each file is accessible only to a particular user at a particular time.

An Internet page – a data file accessible through the Internet – is commonly referred to as a Web page (from World Wide Web page). Web pages are stateless. They do not store therein any data to distinguish one access attempt from another or one user from another. Unfortunately, this means that a web page cannot maintain customer information between page calls, which is critical to modern Internet based applications.

There are two known methods of passing information between web pages: transferring state information between pages via a query string or form fields and using a "cookie." The first approach allows easy tampering of a site - URL - or of form-encoded data. This has been further prevented by encoding an entire web site with SSL, which is very resource consuming. The second approach is more commonly used; an example is Microsoft IIS. A cookie is a piece of information relating to an access session of an individual. The cookie is provided from a server to a client. The client then provides the cookie with each access request. As such, the cookie identifies the user and can be limited to certain destination addresses, certain times, and so forth. The use of cookies has two main disadvantages: When using a web farm, a number of web servers working as a single server, and storage of session variables on a web server within the web farm a problem often arises when a user is redirected to another physical server within the web farm. This is an evident problem since the new server is not the server actually storing the user information for that user. Some problems that arise include a user having to log in again, loss of all instructions to that point, loss of a current shopping cart or its contents and so forth. The second problem is that some browsers provide a page like that shown in Fig. 1 that warns them that the information from the Internet is being stored on their computer. This page is often confusing or alarming to a user possibly resulting in a negative experience with a particular web site. Of course, web site designers and owners do not want to form any negative impressions of their sites anymore than a store owner wants customers to feel uncomfortable in their store.

In United States Patent 5,774,670 in the name of Montulli and issued on June 30, 1998 and United States Patent 5,826,242 in the name of Montulli and issued October 20, 1998, a system for maintaining persistent client state in a hypertext transfer protocol based client-server system. In an embodiment described therein, an http client requests a file, such as an HTML document, on an http server, and the http server transmits the file to the http client. In addition, the http server transmits a state object, which describes certain state information, to the http client. The http client stores the state object, and will typically send the state object back to the http server when making later requests for files on the http server. Such a system suffers drawbacks as noted above. First, when the state information is in the form of a cookie, it must be stored on the client system. Alternatively, anyone intercepting the state information has access to the data within the server.

In United States Patent 5,875,296 in the name of Shi et al. and issued February 23, 1999 a distributed file system web server user authentication method based on cookies is presented. Once again, cookies present a problem in that they are stored on a client system, which is inherently intrusive, and in that messages that are confusing or raise concerns may be displayed to a user of the client system.

In United States Patent 5,907,621 in the name of Bachman et al. and issued on May 25, 1999, a system and method for session management is presented. A secure token is made part of each HTML page sent to a user from the server computer and the token is returned to the server with each submitted transaction request. The token is compared with token information originally sent out. The submitted request is recognised as being from an authorised user if the token and a session table have the same information. Unfortunately, intercepting such a token is possible and as such, security provided with such a token is insufficient for many applications.

In United States Patent 5,884,312 in the name of Dustan et al. and issued on March 16, 1999, a system and method for securely accessing information from disparate data sources through a network is described. The method includes communicating a session identification number and a portion of the logon input to a client. the method further includes requesting a first function by the client and communicating the session identification number, the portion of the logon input, and the first function request to the network server from the client. Next, the session identification number and the portion of the logon input are verified as valid prior to providing access. Unfortunately, this method is no more secure than prior art session identifiers in that intercepting a request results in full access to a given session.

It would be advantageous to provide a method of identifying a session that requires little computational overhead and also provides a reasonable level of security. Preferably, such a method would be capable of supporting time limited sessions in a secure fashion that prevents tampering with a sessions time limits.

It would also be highly advantageous to provide a method of identifying a session that does not require storage of a cookie on a client computer system.

Summary of the Invention

In order to overcome these and other limitations of the prior art it is an object of the present invention to provide a method of identifying a session that requires little computational overhead, provides some level of security, is securely capable of supporting time limited sessions, and does not require storage of a cookie on a client computer.

In accordance with the invention there is a provided a neural network for use in transforming session identification values in a one-way fashion that is not easily reversible or decodable. The use of the neural network allows for inclusion of other parameters such as client address and system time in the encoding process to further secure the overall transform. Also, the neural network is modified periodically in order to change the overall transform and prevent hacking.

In accordance with another embodiment of the invention there is provided a method of identifying a session formed between a client system and a network accessible server system for providing secure access to multiple network accessible secure files comprising the steps of: transmitting from the client to the server a request including a session identifier; transforming the session identifier using a repeatable non-reversible transform to determine a transform value; comparing the transform value to a value associated with the session to provide a comparison result; and, when the comparison result is indicative of a match within predetermined limits, processing the request to provide results and transmitting the results to the client.

Preferably, the value associated with the session is a value previously determined using the repeatable non-reversible transform and is provided from the client with the session identifier. In this way, any server supporting execution of the repeatable non-reversible transform is able to verify the session identifier when provided with the hashed value. This obviates a need for an enterprise database for storing the session identifiers.

In accordance with yet another embodiment of the invention there is provided a method of identifying a session formed between a client system and a network accessible server system for providing secure access to multiple network accessible secure files comprising the steps of: commencing an access session for accessing files on a server in communication with a wide area

network; generating a number for use in identifying the access session; hashing the generated number; storing the hashed generated number on an enterprise server forming part of the enterprise of which the at least a server forms part; and, providing at the generated number to the client computer for provision to the server in order to identify the session.

An advantage of the present invention is that because the transform is not a mathematical transformation, it is very difficult to decode. A further advantage is that the transform is repeatably performable given a same set of input values. Further advantages will become apparent from the descriptions of the embodiments set out below.

Brief Description of the Drawings

Exemplary embodiments of the invention will now be described in conjunction with the following drawings, in which:

Fig. 1 is a typical warning message provided by a commonly available Internet browser when cookie data is received from a server;

Fig. 2 is a simplified flow diagram of a general embodiment of the invention;

Fig. 3 is a simplified flow diagram of another embodiment of the invention; and,

Fig. 4 is a simplified data flow diagram of a neural network transforming data values provided thereto into output data values.

Detailed Description of the invention

The following terms are used in this application and have the following meanings:

Network: a wide area network for electronic communication between a plurality of computers each computer in communication with the network;

User: a person using a computer;

Client: a computer system in communication with a communications network and making a request of another computer system in communication with a same communications network;

Server: a computer system in communication with a communications network and receiving, processing and responding to a request from a client computer system in communication with a same communications network;

Enterprise database: a database for serving computers in communication therewith and forming part of a same enterprise; and,

Session: a grouping of requests from a same client and their associated responses. Typically a session relates to a single matter such as shopping in a single electronic store or performing stock trading. Commonly, sessions are timed to prevent a session from remaining open after the client has ceased communications. As noted above, currently sessions are managed using encryption keys, cookies, or other identifiers. Each of the known methods is problematic.

According to the invention, a method of identifying a session is provided. According to the method information that is unique to a particular session is encoded according to a repeatable non-reversible operation such that the server can regenerate the information repeatably given equivalent input data. Preferably, the data is time stamped to allow for an expiration of the data or other time related processing. By including an identifier and data relating to the client within the information, tampering is made difficult since additional information is necessary to intercept the data. Further by including time data, a session duration is determinable in a manner which is very difficult to tamper with. Optionally, the encoded data is stored in an enterprise database such that the enterprise server can regenerate the information repeatably given equivalent input data and compare the regenerated data to the stored data. Alternatively, the client is provided with the encoded data and the unencoded data and, as such, any server having a same encoding algorithm can verify the validity of a provided session identifier and encoded value pair.

Referring to Fig. 2, a simplified flow diagram of a general embodiment of the invention is shown. Information in the form of a unique session identifier is stored in an enterprise database. Preferably the unique session identifier comprises a unique long number determined for each particular session. To ensure the unique long number is not tampered with, a hash-code based on the unique long number is generated within the enterprise database. The hash code is secured using other session parameters such as a current server system time, browser software version, and client IP address. Hashing operations are repeatable in that the operation, given a same set of input values, always results in a same result. In this fashion, the hashing result is capable of being later verified against a unique session identifier passed with a query string from the client to the server. Further because other session parameters are used in the hashing process, the verification is further dependent on the accuracy of these parameters. This prevents

fraudulent access to the web pages by a third party. For example, credit data from a trading account may be available on a web page for anyone with a correct password. Because the credit data is presented on a number of different pages, when accessed, a session is begun. Anyone with the session identifier can now access that session and therefore view the data. However, only data on a particular server can be viewed since only that server has the session identifier stored thereon. Because, according to the invention, the session identifier is hashed along with other session related data, it is assumed that only requests from a same client computer and providing a correct session identifier will be acceptable. Further, when the hashed value is provided along with the session identifier, any server executing a same hashing algorithm can verify the session identifier. To prevent unauthorised access to the data, a unique session identifier provided from a client computer and passed with query string is compared to the previously generated hashed value for the same session. When a match exists, the client is assumed to be a same authorised client for a current session. The previously generated hashed value is preferably provided to the client when generated and the provided from the client to the server with the session identifier for verification. Alternatively, the previously generated value is stored in an enterprise database.

Referring to Fig. 3, a simplified flow diagram of another embodiment of the invention is shown. The method comprises a step of performing a one-way hash to encode a session identifier in the form of a database key passed with a query string. In performing the one-way hash operation, data indicative of the client system is used in the form of an IP address and/or a browser version. This prevents other client systems from sending a same database key for access to the session once started since, when hashed, the value that results is different since the IP address of the other client system is different from that of the client system. Preferably, each value is calculated using a portion of the server current system time and data relating to a user browser type. In this way, it is difficult to tamper with the resulting code since a user attempting illegal access requires data relating to the actual client system and software as well as the hashing algorithm. The use of the server current time allows for a session to be timed to expire in a fashion that is extremely secure since tampering with the session time is impossible without access to the hashing algorithm.

Because the hash operation is a one-way operation, it is impossible to determine each of the necessary values from the code stored in the enterprise database and/or provided to the client and, as such, the hashed values need not be secured. Further, when knowledge of the hash algorithm is guarded - it is kept secret, the codes may be stored in a fully accessible location or transmitted over unsecure communication paths since other users tampering with values results in unusable sessions but not in access to secure data. As such, data integrity and security is not compromised by providing access to the values. However, for user convenience, it is preferred that the values in the enterprise database are stored in a reasonably secure manner from access and tampering.

Of course many values can be used to identify a user, a client system and so forth. For example, the internet provider with which a client is directly communicating may be used as a value included in the hashing algorithm, the operating system of the client system may be used as a variable in the hashing algorithm, the user's name, authorisation code and/or identifier may also be used, and so forth. Of course, the data described above with respect to the preferred embodiment were selected because they are already provided to a server by a client system and, therefore, such a method incurs no additional overhead in data transfer and requires no modifications to current client systems. For example, each browser name/version combination is mapped to a unique integer value that is provided to a server upon issuing a file access request. A server receives each requestor's IP address in order to ensure that response data, sent to the requestor in response to a request, is directed toward the client systems actual location.

Since the security level of a method according to the present invention is based on the hash algorithm, a novel implementation of a hash algorithm using neural networks is presented below and forms inventive subject matter for the present invention.

Trained neural network in producing hash-code

One-way hashing is a technique of encoding information, x_1, for transmission across a public network using a hashing algorithm A(). Applying the hashing algorithm A() to the information, x_1, results in y_1 which is also written as $A(x_1)$. The transformed data y_1 is then stored for later use in verifying session identifiers. The information x_1 is transmitted to the client to identify the session. When a request is received, a session identifier x_2 is provided therewith.

This session identifier, x_2, is then encoded as $y_2 = A(x_2)$ and compared to y_1. When the two values match, the session identifier is verified. If there is no match the session identifier was tampered with, other data provided to the hashing algorithm, A(), is changed, or the stored value has been modified. The weakness of the method according to the invention is that if somebody knows or guesses the algorithm A() then that individual is able to produce a valid hash-code. Most of the hash-code algorithms are known to hackers, which makes the task of guessing A() even easier.

An innovation of the present invention is to either slightly change one of the known hashing algorithms by substituting a neural network trained to reproduce most of this algorithms results instead of the traditional implementation or make to design a hashing algorithm that changes dynamically. Dynamically changing an algorithm can be performed by means of changing a neural network structure or link weights within the neural network. Also, dynamically changing the hashing algorithm can be accomplished by producing new neural networks based on selected or random training data. In either case, it is much more difficult to guess a hashing result or algorithm when a simple mathematical transform is avoided.

Using a neural network for performing hashing functions

A neural network is a mapping processor that is believed to replicate some brain functionality. A hardware based neural network receives a set of inputs, each input at an input port, and a provides a set of outputs, each output at an output port. Essentially, the processor is trained by providing inputs and desired outputs. Within the processor a plurality of weighted links exist. During training, the weights are adjusted to best reflect the transform from input data set to desired output data set. Because the correlations are not related to a predetermined mathematical function, the resulting system only approximates a function on which the training data is based. Therefore, using such a system allows for a predictable transformation - one that is repeatable - that is likely difficult to decode since it is not necessarily mathematically consistent.

A software based neural network is very similar though instead of input ports it receives input values and provides output values. Essentially, neural networks are popularly used in correlation processing - where inputs and outputs are known but a function relating the two is either too complex to be computationally efficient or too difficult to discern.

According to the invention, a neural network is used for generating output values $(y`_1,y`_2,..,y`_n) = N(x_1,x_2,..,x_n)$ instead of the actual algorithm $(y_1,y_2,..,y_n) = A(x_1,x_2,..,x_n)$. The input/output pairs (x_n, y_n) produced by the algorithm A() are used as training data for the neural network. Batch Back Propagation is a preferred learning method for this purpose as it introduces the higher degree of generalisation to the resulting neural network when compare with original hash algorithm. Batch Back Propagation is a known neural network training method relying on back-propagation with weight updates occurring after each epoch for training. Of course, other neural network training methods are also useful in implementing the present invention.

A neural network, because it is not a mathematical transform, is difficult to predict absent the actual trained network to execute. Therefore, the above method is highly advantageous over prior art hashing techniques. That said, further advantages flow from modifying the neural network periodically in order to further reduce any risk of predictable transform results.

Referring to Fig. 4, a simplified flow diagram of hashing performed by a neural network 10 is shown. An input value h is provided to the neural network and an output value H results. When the neural network undergoes further training, the transform performed thereby changes. Therefore, the same input h produces a new output H`. The new output H` may, in fact, have an exact same value. Alternatively, the new output H` may have a different value. This depends on the training data provided and the learning process of the neural network 10.

Modifying network structure and/or links weight

The neural network described above may be dynamically altered to produce corresponding changes in the underlying transform to $y`_n$. For example, the weights of even neurons of layer 3 within the neural network may be incremented by 0.01. Preferably, these changes are done periodically based on the current system time resulting in a predictable change to the neural network. Then, by establishing at what time a session identification was encoded, an appropriate neural network is used when re-encoding to result in a same value given identical transform input values. Alternatively, all session identifiers are re-determined whenever a neural network modification occurs.

According to another embodiment, each decimal digit of a numeric value is encoded as a character string of length n. The encoding is done using a neural network trained on a predetermined training data set. For example, for n = 3 the following table was used to train the network that has four inputs – current decimal digit of 0..9 (input 1 - production input), user IP address (input 2), server current system time (input 3) and user browser type (input 4) and 3 outputs. Inputs 2, 3, and 4 are optional and are used depending on desired strength of a resulting security system. The number of additional inputs may also vary.

Production input	Additional inputs			Outputs		
Input 1	Input 2	...	Input n	Output 1	Output 2	Output 3
1	Specially selected values			"A"	"S"	"t"
2				"M"	"j"	"L"
...				Different combination of symbols		
9				"g"	"v"	"T"

The integer 19834, for example, was encoded as the character string [AnJPLOY0jNNmewW]. If the length of the unique session identifier, a number of digits within the identifier, is less then 6 digits the value of the unique session identifier is subtracted from 2147483648 and the result is encoded. This maintains some standardisation of encoded value length.

For encoding of long data sequences such as text messages, each group of n input bytes is encoded into a single output value using a method similar to that above. For example, the text "table " may be encoded as 71 if the network has 3 production inputs and one output. The following training table illustrates the basics of this embodiment of the method:

Production inputs			Additional inputs			Output
Input 1	Input 2	Input 3	Input 4	...	Input n	Output 1
"t"	"a"	"b"	Specially selected values			7
"l"	"e"	" "				1
Training data						Selected numbers

Alternatively, the hash code is generated other than in the enterprise database and is provided to the enterprise database for storage. Further alternatively, the hash code is generated on any server in execution of the hashing method and is provided to the client. When a single server is implemented, it is a simple matter to maintain the enterprise database on the same server and to process the steps of the hashing method according to the invention on the same server. Alternatively, when a server farm is used, the enterprise database is on a server within the server farm, or alternatively, on another server accessible to the web farm. Of course, when the hashed code is provided to the client, no enterprise database is necessary. The hashing algorithm may be performed on any accessible system. Preferably, the system providing execution of the steps of the hashing method is secure from tampering.

Because the session identifier is encoded and stored within an enterprise database, sessions are maintainable across different servers within a web farm, different servers within an enterprise, and potentially disparate servers located remote one form the other and forming part of different enterprises wherein each has access to the hashing/encoding system.

Accordingly, the invention provides a secure session identifier for use in maintaining a session having a limited or unlimited duration that is functional across different physical server systems and that does not require storage of a "cookie" on a client system. Further, the invention allows for implementation completely on the server system using existing client system functionality.

Appendix A has a listing of source code for software implementation of an embodiment of the present invention.

Numerous other embodiments of the invention may be envisaged without departing from the spirit or scope of the invention.

Claims

What is claimed is:

1. A method of identifying a session formed between a client system and a network accessible server for providing secure access to multiple network accessible secure files comprising the steps of:

a) transmitting from the client system to the server a request including a session identifier;

b) transforming the session identifier using a repeatable non-reversible transform to determine a transform value;

c) comparing the transform value to a previously determined value associated with the session to provide a comparison result; and,

d) when the comparison result is indicative of a match within predetermined limits, processing the request to provide results and transmitting the results to the client system.

2. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 1 wherein the previously determined value is a value provided from the client system with the session identifier, the value determined by the step of:

transforming the session identifier using the repeatable non-reversible transform to determine a determined value.

3. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 2 wherein the steps of transforming the session identifier are performed by at least a server, the at least a server using a same transformation and including the step of:

providing to the client system the determined value.

4. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 1 wherein the previously determined value is a value stored within an enterprise database, the value determined by the step of:

transforming the session identifier using the repeatable non-reversible transform to determine a determined value.

5. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 4 wherein the steps of transforming the session identifier are performed by at least a server, the at least a server using a same transformation.

6. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 1 wherein the step of transforming the session identifier is performed in dependence upon further data provided from the client system and relating to the client system.

7. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 6 wherein the further data comprises data unique to the client system at least at the time of the request.

8. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 7 wherein the further data comprises an IP address of the client system.

9. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 6 wherein the further data comprises data relating to software in execution on the client system.

10. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 1 wherein the step of transforming the session identifier is performed in dependence upon time data.

11. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 10 wherein the time data is determined based on a session time for the session, the session time stored in association with the stored value.

12. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 11 wherein the time data is a portion of a system time for a server performing the transform at a time approximately when the session is initiated.

13. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 1 wherein the step of transforming the session identifier is performed according to a transform that is modified at intervals.

14. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 13 wherein the step of transforming the session identifier is performed according to a transform that is modified at intervals in a predictable fashion such that given a time of a previous transform and any input values thereto, a same transform output is produced even after modification of the transform.

15. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 1 wherein the step of transforming the session identifier includes the following steps:
providing the session identifier to a neural network as at least an input value thereto; and,
providing at least an output of the neural network as the transformed session identifier.

16. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 15 wherein the neural network is modified at intervals.

17. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 16 wherein the neural network is modified at intervals in a predictable fashion such that given a time of a previous transform and any input values thereto, a same transform output is produced even after modification of the transform.

18. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 16 wherein the neural network is modified at intervals by the steps of:

providing a new training data set to the neural network; and,

performing the step of training the neural network using the new training data set.

19. A method of identifying a session formed between a client system and a network accessible server system for providing secure access to multiple network accessible secure files comprising the steps of:

a) commencing an access session for accessing files on at least a server in communication with a wide area network;

b) generating a session identifier for use in identifying the access session;

c) hashing the session identifier for use in verifying the session identifier when provided from the client system;

d) providing the session identifier to the client system for provision to the server in order to identify the session.

20. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 19 comprising the step of

e) storing the hashed session identifier on an enterprise server forming part of the enterprise of which the at least a server forms part.

21. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 19 comprising the step of

e) providing the hashed session identifier to the client system for provision to the server with the session identifier in order to verify the session identifier.

22. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 19 wherein the step of hashing the session identifier is performed in dependence upon further data provided from the client system and relating to the client system.

23. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 22 wherein the step of hashing the session identifier is performed using a neural network accessible to the at least a server.

24. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 23 wherein the further data comprises an IP address of the client system.

25. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 23 wherein the step of hashing the session identifier is performed in dependence upon time data.

26. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 25 wherein the time data is determined based on a session time for the session, the session time stored in association with the stored value.

27. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 23 wherein the neural network is automatically modified at intervals.

28. A method of identifying a session formed between a client system and a network accessible server system for providing secure access to multiple network accessible secure files comprising the steps of:

a) commencing an access session for accessing files on at least a server in communication with a wide area network;

b) generating a session identification number for use in identifying the access session;

c) providing the session identification number and at least another value related to a session parameter to a neural network as input values thereto;

d) receiving values provided as output values from the neural network in response to the input values provided to the neural network; and,

e) providing at the session identification number to the client system for provision to the at least a server in order to identify the session.

29. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 28 comprising the step of
e) storing the received values on an enterprise server forming part of the enterprise of which the at least a server forms part.

30. A method of identifying a session formed between a client system and a network accessible server system as defined in claim 28 comprising the step of
e) providing the received values to the client system for provision to the server with the session identification number in order to verify the session identification number.

31. A method of identifying a session as defined in claim 28 wherein the neural network is trained using Batch Back Propagation.

32. A method of identifying a session as defined in claim 28 wherein the neural network is altered at intervals automatically in order to vary some output values resulting from some input values.

33. A method of identifying a session as defined in claim 32 wherein the neural network is modified at intervals in a predictable and reversible fashion in order to vary some output values resulting from some input values.

34. A method of identifying a session as defined in claim 33 wherein the reversible fashion is a predictable modification based on time such that given an input value and a time, a same set of output values results.

35. A method of identifying a session as defined in claim 28 wherein the other value comprises data provided from the client system and relating to the client system.

36. A method of identifying a session as defined in claim 35 wherein the data provided from the client system and relating to the client system includes an IP address of the client system.

37. A method of identifying a session as defined in claim 35 wherein the data provided from the client system and relating to the client system includes data relating to software in execution on the client system.

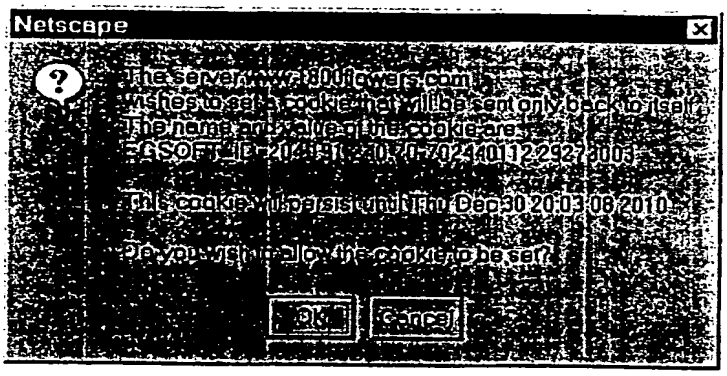


Fig. 1

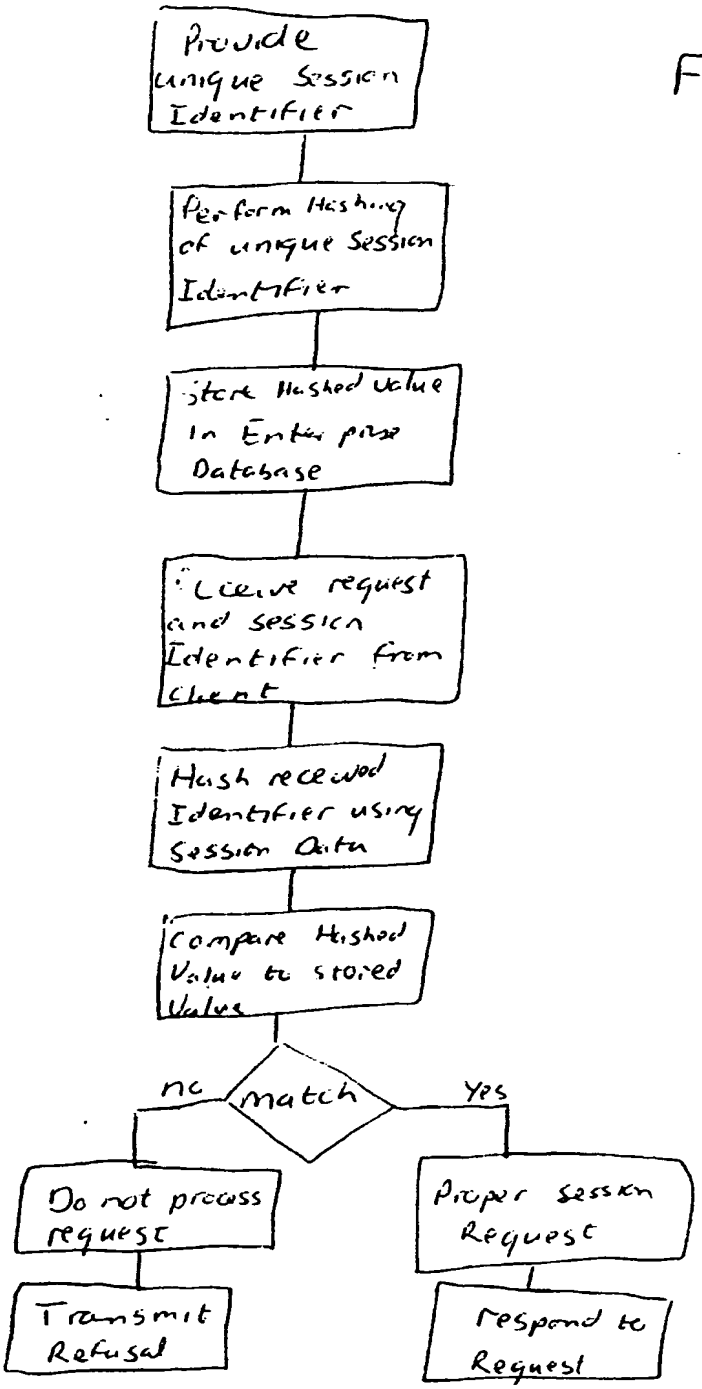


Fig. 2

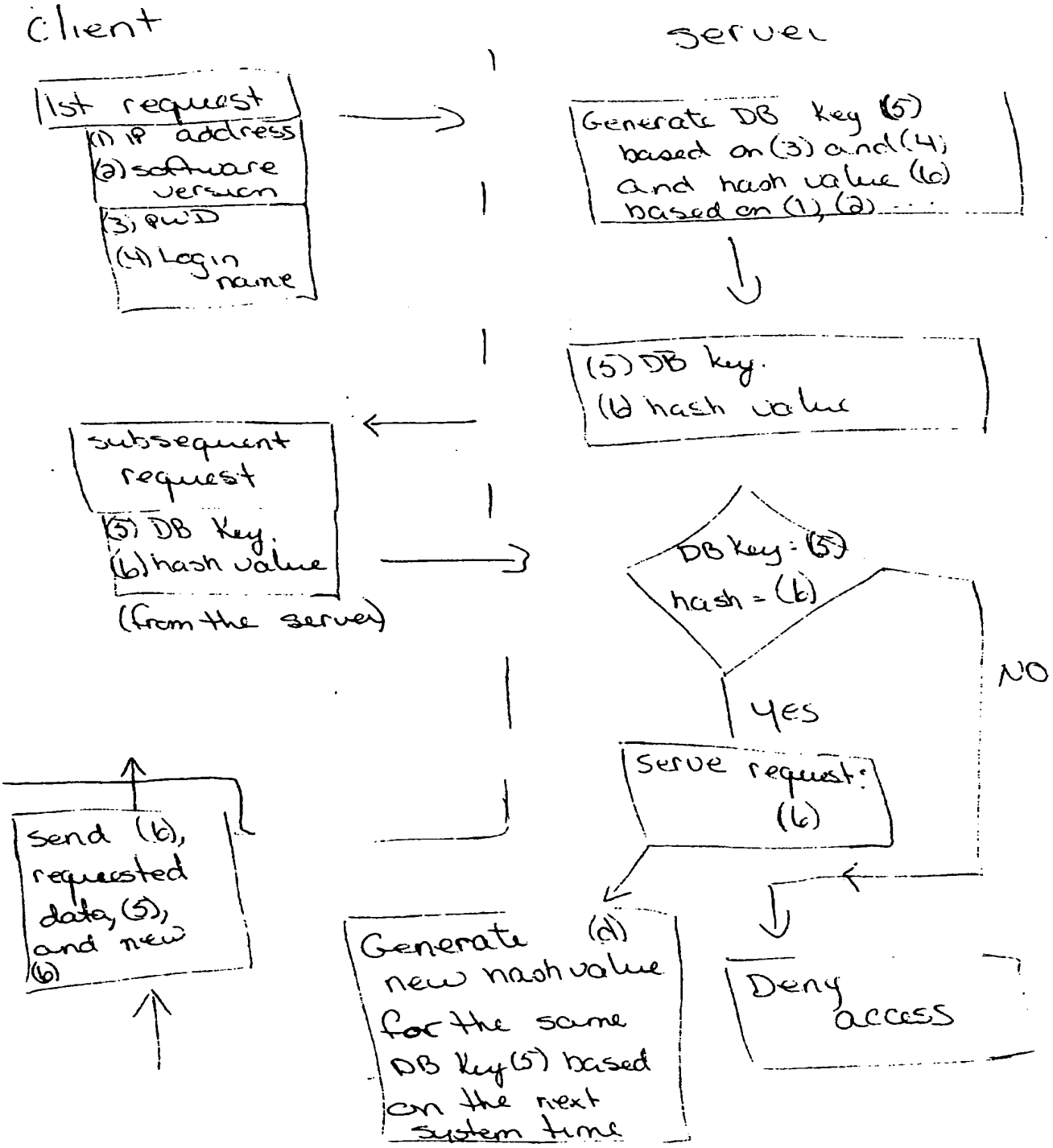


Fig 3

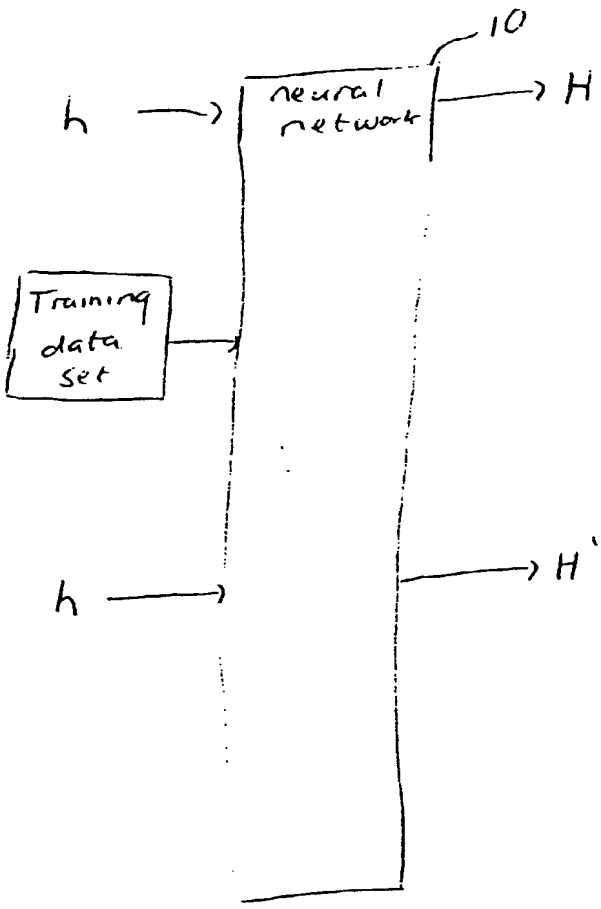


Fig. 4